Compliance checking in the energy domain via W3C standards *

Joseph K. Anim[1], Livio Robaldo[1], and Adam Wyner[1]

Swansea University, Swansea, UK
{joseph.anim,livio.robaldo,a.z.wyner}@swansea.ac.uk

Abstract. This paper investigates the use of W3C standards, specifically RDF, OWL, SPARQL, and SHACL, to automate legal compliance checking. We consider in particular in-force regulations for extracting oil and gas in Ghana to exemplify our proposed model. This paper models some selected norms from these regulations into a sample RDF ontology along with inference rules to check their compliance with respect to a given state of affairs. The paper's main finding is that inferences enabled by OWL and SHACL shapes are not expressive enough to represent some existing legal requirements, specifically those imposing constraints on metadata about RDF individuals. To achieve the required expressivity, it is proposed that SHACL-SPARQL rules should be instead used.

Keywords: Compliance checking · W3C standards · symbolic AI

1 Introduction

Due to the ever-growing regulations upon which compliance procedures are conducted, the quantum of documents that companies must submit to prove their compliance with the regulations governing their activities has increased and is increasing in volume[1]. In addition, lawyers mostly check compliance and prepare due diligence documents manually. However, this has several disadvantages: it is highly time-consuming, it is error-prone, and it creates an avenue for corruption as it makes it difficult to understand when errors were either caused by unintentional oversights or they were done on purpose.

LegalTech technologies aim at mitigating these problems [4]. Automatizing repetitive operations allows one to save time, enhances accuracy, and makes the whole process easily accountable, which in turn makes corruption less feasible.

Currently, most approaches to LegalTech are based on Machine Learning (ML), see, e.g., [5,17,31,15,14]. However, ML makes it difficult to handle *specific and exact* values, as it is often required when checking compliance of due diligence documents as well as drawing inferences from the values. The accuracy of ML

* Joseph K. Anim has been supported by the Ghana Scholarship Secretariat (see https://www.scholarshipgh.com).

[1] https://www.thomsonreuters.com/en-us/posts/investigation-fraud-and-risk/cost-of-compliance-2021

is intrinsically limited by the Pareto principle, a.k.a., the 80/20 rule; thus it provides results that are correct in *most* cases but not *all* [23].

Furthermore, ML tends to behave like a "black box" unable to explain its decisions; as a consequence, often it is even even impossible to explain the difference between the ∼80% of correct results from the ∼20% of incorrect ones.

To address these limitations, symbolic representations and rules have been proposed and used even though they require more manual efforts. Symbols correspond to human-understandable concepts and rules represent how we reason with the symbols, thus the chain of logical derivations on symbols could provide intelligible explanations of AI decision-making. To mitigate the fact that symbolic representations and rules require more manual efforts, *standardized formats should be used*, thus allowing the funnelling of efforts from more people, which in turn facilitates reusability and sharing of resources.

This paper presents a methodology for compliance checking based on main W3C standards for the Semantic Web. The methodology is exemplified on Ghanaian regulations for extraction of oil and gas, which we will use as case study.

We believe that it is crucial to research and implement symbolic compliance checkers that are compatible with the mentioned W3C standards because more and more (big) data is becoming available in RDF format. Matching and annotating big data with legislative information will produce even more and richer big data. Thus, the importance of using the same standardized formats, namely the W3C standards, to achieve interoperability.

In addition, legal ontologies encoded in RDF/OWL have been increasingly proposed and used within existing LegalTech applications [18,16,28,26]. Legal ontologies specify relevant legal concepts, individuals, constraints, etc. such as duties and rights from legislation, as well as their relationship with the concepts of the domain to which the norms apply, e.g., finance, health, or the energy domain.

By using legal ontologies, states of affairs need to be checked against the constrains of the ontology in order to check for compliance. The present paper proposes a novel approach to compliance checking using SHACL-SPARQL rules, discussed below. In contrast to prior work, e.g., [10] and [27], which respectively use OWL restrictions and SHACL shapes together with SHACL Triple rules, the current work uses SHACL-SPARQL rules which are able to extract metadata, aggregate it, then perform some process on the aggregated information. This is not feasible via OWL restrictions, SHACL shapes, or SHACL Triple rules. The framework is exercised with respect to an ontology representing regulations in the oil and gas domain.

2 Background - W3C standards for the Semantic Web

As mentioned earlier, the objective of this paper and of our research as a whole is to devise computational methods for compliance checking fully compatible with main W3C standards, in order to foster interoperability with available big data.

W3C has already defined several formats to empower the Semantic Web[2]. This paper will use RDF and OWL to encode the ontology, both the TBox (terminological box), which represents the domain knowledge, and the ABox (assertive box), which represents the states of affairs.

Then, we will use SPARQL and SHACL to compute and query new knowledge from the explicitly asserted RDF triples. Specifically, we will model norms as SHACL-SPARQL rules; these rules will then be executed on the states of affairs encoded in RDF to infer which individuals comply with the norms rather than violating them.

2.1 RDF and OWL

RDF (Resource Description Framework) is a language which represents information about resources on the World Wide Web. RDF represents metadata about web resources, such as the title, author, and modification date of a web page, etc. However, RDF has evolved such that it can be used to represent any general information about identifiable things on the web. The intent behind RDF was to allow applications to process and exchange information without this information losing its intended meaning. This will ensure that information can be exchanged even between applications that were not developed to use or work with the original information.

RDF is therefore used for creating ontologies, i.e., application-neutral networks of concepts, which are called "RDF resources" (classes, individuals, and properties). RDF includes basic constructs to declare them as well as to relate them to one another.

RDF has been mainly designed to *describe* knowledge. Thus, it has very limited reasoning capabilities. In RDF, it is only possible to infer whether certain RDF resources belong to certain classes via the constructs `rdfs:subClassOf`, `rdfs:domain`, and `rdfs:range`.

OWL (Ontology Web Language) augments RDF by adding more reasoning capabilities. OWL allows specification of many more constraints than RDF, which in turn enable more inferences about the RDF resources. In particular, OWL introduces constructs that allow to infer when certain RDF resources do *not* belong to certain classes, e.g., the construct `owl:disjointWith`. These constructs in particular amplify the reasoning capabilities of the language, which in turn has led to investigations about the trade-off between expressivity and computational complexity of the inferences.

These investigations have identified three main sub-languages of OWL: OWL full, OWL DL, and OWL lite. OWL full and OWL lite feature, respectively, full and very reduced expressivity but, consequently, also full and very reduced computational complexity. OWL DL has intermediate expressivity and complexity between OWL full and OWL lite; DL stands for "Description Logic", the logic

[2] See the list at https://www.w3.org/2001/sw/wiki/Main_Page

that OWL DL refers to[3]. Thus, for applications in which the computational time is relevant, it is advisable to use OWL DL or OWL lite in place of OWL full.

Several OWL reasoners have been already proposed in the literature to compute inferred ontologies from the explicitly asserted one, e.g., HermiT[4]. [9] presents a comparison among some of these OWL reasoners. [9] highlights that the reasoners vary significantly with regard to the relevant aims and characteristics, so that each specific case study, especially if in an industrial context, deserves a careful and critical choice of the reasoner to be employed.

2.2 SPARQL and SHACL

As part of the Semantic Web activity, the RDF Data Access Working Group released in 2004 the first public working draft of an RDF querying language which was known as SPARQL. Since then, further operators to add, delete, and update the triples in the ontology as well as to deduce new information from them has been added to SPARQL. Nowadays SPARQL is a rich language for both querying and manipulating RDF datasets [21].

SPARQL query are generally embedded and executed within other software or programming languages; examples are the SPARQL plug-in for the Protégé editor and the Jena libraries for Java. Therefore, the order in which SPARQL queries are executed is decided by the user or programmatically in the logic of the software: SPARQL does not provide constructs to relate the queries of one to another, for instance, to establish some execution order on them.

On the other hand, SHACL is a W3C recommendation more recent than SPARQL: it was originally proposed in 2017 for the purpose of validating RDF datasets. SHACL allows to specify special constraints, called "SHACL shapes" on RDF resources. External validators allow to check whether an RDF dataset is valid or not with respect to a set of SHACL shapes. SHACL is more expressive than OWL, and it may be therefore used to augment the inferential capacities. In particular, SHACL includes non-monotonic operators such as negation-as-failure. These are not allowed in OWL, which is a monotone language.

Furthermore, SHACL constraints are more flexible and easier to edit than OWL ones because, while the latter are all executed *at once*, in SHACL we may *decouple* complex validation tasks into (simpler) sequential modules. This is possible thanks to the introduction of SHACL rules[5] that enable non-ontological types of operations such as collecting data from RDF resources located in "distant" parts of the ontology or computing partial results needed for the validation [20]. SHACL allows in particular to specify *priorities* on the rules, and so to define sequences or even flow charts of rules, in a rather controlled fashion.

There are two kinds of SHACL rules: SHACL Triple rules, which can add a *single* RDF triple to the inferred ontology, and SHACL-SPARQL rules, which

[3] Description Logic refers to a *family* of logics that are less expressive than First-order Logic; OWL DL more specifically refers to the description logic **SHOIN-D** [13].

[4] http://www.hermit-reasoner.com

[5] https://www.w3.org/TR/shacl-af

embed SPARQL queries in the form `CONSTRUCT-WHERE`. In SHACL-SPARQL rules, for each subgraph that satisfies the `WHERE` clause, the subgraph in the corresponding `CONSTRUCT` clause is added to the inferred ontology. Therefore, contrary to SHACL Triple rules, SHACL-SPARQL rules may add more than one RDF triple to the inferred ontology. Moreover, the expressivity of SHACL-SPARQL rules add to the richness of SPARQL the possibility of establishing *orders* between SPARQL inferences, thus creating the controlled sequences or flow charts of such inferences.

3 Related works

As explained in the Introduction, approaches in LegalTech (indeed, in AI in general) may be classified in two main categories: approaches based on statistical reasoning, i.e., Machine Learning (ML), and approaches base on logical reasoning, such as the one proposed in this paper.

ML approaches are predominant, also for compliance checking; examples are [31] and [15]. In these approaches, ML is used to detect anomalies, i.e., behaviours or outcomes that diverge from the general trend identified statistically.

For instance, [15] observed that, in current real-world Enterprise Resource Planning (ERP) systems, the rules for VAT compliance are mostly maintained manually by ERP VAT experts, due to the large amount of variables involved, which are in turn subject to changes in regulations, laws, tax rates, or internal business strategies and preference criteria of the different industries and companies. Therefore, it is rather easy for the ERP VAT experts to make mistakes (see [15]). In light of this, [15] developed a supervised learning classifier for the VAT tax code determination process able to identify and notify anomalies in the results of the calculations.

The objective of our approach differs substantially from the one of [15] or similar anomaly detection ML-based solutions. Our research aims at defining a standardized methodology for encoding the if-then rules coming from the in-force regulations or the companies' internal business strategies. In addition, we envision a future in which the if-then rules coming from the in-force regulations are directly provided by the appointed public authorities, rather than being (re)encoded by each company, each in the specific format of the ERP system used. Consequently, each company will only have to encode the if-then rules related to the company's internal business strategies and preference criteria. In this scenario, ML-based algorithms such as [15]'s can be still used to detect anomalies: the two solutions are fully orthogonal.

In our view, the standardized methodology that we advocate should be defined in terms of the W3C standards RDF and SHACL, both because they provide the required expressivity, as this paper will show, and because they are at the basis of the World Wide Web, i.e., they are widely used worldwide and they enable interoperability with external available knowledge bases, which could lead to higher coverage and accuracy of the applications.

However, the present paper is not the first attempt to check compliance on RDF data. Some of the first approaches are [12,7,8,19]. These approaches use RDF/OWL to model the TBox while the states of affairs and separate knowledge bases of rules are encoded in special *separated* XML formats such as SWRL[12], LKIF-rules [7], RuleML [8] and LegalRuleML [19].

More recently, but in the same vein, [11] made a preliminary proposal to extend the LegalRuleML meta model [3] and to represent normative rules via SPARQL queries. [11] is, to our knowledge, the first proposal that models normative reasoning by employing W3C standards only. The solution presented in this paper is therefore rather close to [11]'s; however, while [11] only uses SPARQL, our formalization will use SHACL in conjunction with SPARQL.

Another relevant approach is [6], which encodes legal rules within OWL2 decidable profiles in order to keep computational complexity under control. In [6], norms are represented as property restrictions that refer to the subsets of individuals that comply with the norms. Compliance checking is then enforced via OWL2 subsumption. However, the authors themselves acknowledge (see [6], §3.3) that their approach does not really involve legal reasoning, which is defeasible in nature, and it is only limited to GDPR policy validation.

Similarly to [6], [10] and [27] distinguish compliant and non-compliant individuals by introducing, respectively, special OWL subclasses and special SHACL shapes. Contrary to [6], however, [10] and [27] can model defeasible inferences.

For instance, the OWL ontology in [10] include two classes `Supplier` and `Vehicle`. The individuals in `Supplier` are obliged to communicate their contractual conditions to their consumers (R1), while vehicles cannot drive over 90 km/h (R2). To implement R1 and R2, `Supplier` and `Vehicle` respectively include a boolean datatype property `hasCommunicatedConditions` and a float datatype property `hasDrivingSpeed`. Then, two subclasses `SupplierR1compliant` and `VehicleR2compliant` are defined, the former including individuals in `Supplier` for which `hasCommunicatedConditions` is true, the latter including individuals in `Vehicle` for which `hasDrivingSpeed` is lower than 90. Compliance checking is then enforced by simply applying OWL2 subsumption. In other words, OWL "is-a" inferences will populate `SupplierR1compliant` and `VehicleR2compliant` with only the individuals that comply with the two norms.

In this setting, exceptions may be added by defining complement subclasses via the OWL2 tags at disposal, e.g., `owl:disjointWith`. These subclasses will define the subsets of individuals that *violate* the norms; thus, by imposing the set the individuals that comply with a norm as `owl:disjointWith` the one that violate it, the correct inferences are achieved.

The solution in [27] is very close to the one of [10], the crucial difference being that [27] uses SHACL shapes in place of OWL2 subclasses/restrictions to validate the values of the relevant attributes, e.g., `hasCommunicatedConditions` and `hasDrivingSpeed`, in the example above. In [27], SHACL Triple rules are used to compute the values of these attributes. These rules collect partial data from the RDF triples, and so they facilitate the representation of the norms by decoupling it in multiple sequential steps. Once the SHACL Triple rules have

computed the values of the attributes, the SHACL shapes are executed in order to validate these values. Individuals with invalid attribute values are labelled as non-compliant. Finally, defeasibility is modeled via negation-as-failure: whenever exceptions hold, corresponding SHACL Triple rules defeat other ones, so that the inferences associated with the latter are blocked.

Finally, very recently [26] propose to efficiently check compliance of RDF data via the DLV2 reasoner [2], which allows for the embedding of SPARQL queries within Answer Set Programming (ASP) clauses. The results in [26] shows that DLV2 can process data much faster than available libraries for SHACL.

The next sections will highlight that the approaches in [11], [10], [27], and [26] are all inadequate, as the expressivity of the underlying formats does not suffice to represent certain kinds of norm that we may find in existing legislation.

Specifically, [10] and [27] are unable to handle compliance on *aggregate data* from the ontology, which are indeed *metadata* about the individuals in the ABox. For example, as described below, we would like to extract metadata (i.e., not specified per se by the ontology) such as the number of Ghanaian technical core employees at a company and the number of all technical core employees at that company, bring them together (aggregate), then use them to *calculate* whether the former is at least 20% of the latter, as required by the regulation.

Aggregate data cannot be computed via OWL "is-a" inferences or via SHACL shapes and Triple rules. On the other hand, to compute aggregate data we need the expressivity offered by SPARQL.

After aggregate data are calculated, we need to compute proportions among them, e.g., 20%, and make mathematical comparisons. In order to execute these computations in the right order (first the aggregate data, then the proportions, and then the mathematical comparisons), we will use the *priority operator* provided by SHACL. No equivalent priority operators are provided in SPARQL or ASP, that is why the frameworks in [11] and [26] are also inadequate.

In light of this, this paper will propose a novel revision of the framework in [27] in which SHACL shapes and Triple rules are replaced by SHACL-SPARQL rules in order to extract metadata, aggregate it, then perform some process on the aggregated information. The ontology and SHACL-SPARQL rules are used in a compliance checking framework depicted in Figure 1.

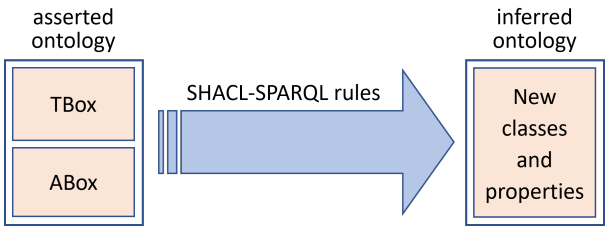

Fig. 1. The compliance checking framework

4 Case study: extracting oil and gas in Ghana

In Ghana, the oil and gas upstream industry has recently seen a lot of foreign investments from international corporations. It is expected these investments will further grow in the near future, due to the conflict in Ukraine and the consequent need for new alternative sources of oil and gas. The explorations have led to several oil and gas discoveries[6], among which it is worth mentioning the Offshore Cape Three Points (OCTP), which is estimated to hold about 41 billion cubic meters of non-associated gas and 500 million barrels of oil.

These discoveries are in turn expected to greatly contribute to the Ghanaian gross domestic product. It has been estimated that the oil and gas industry will contribute approximately 15.94 billion GHS (around 2.76 billion U.S. dollars) to Ghana's gross domestic product in 2024 [29].

As a result, the upstream oil and gas industry has become one of a heavily regulated sector in Ghana. Companies operating in this domain are required to submit several due diligence documents to auditing agencies such as the Ghana Petroleum Commission in order to check for regulatory compliance with respect to the in-force regulations[7]. In this work, we focus on the Local Content and Local Participation Regulations L.I 2204, henceforth named as "L.I 2204" only.

The L.I 2204 aims at ensuring the participation of Ghanaians and the use of indigenous materials in the upstream oil and gas industry. Simply put, the L.I 2204 is intended to prevent foreign companies from bringing their own employees and materials from abroad. Rather, they are allowed to extract the country's oil and gas only on condition they create employment and other economical benefits for the local population.

In particular, the regulation 7(2)(B) of the L.I 2204 requires companies in the upstream oil and gas industry to provide annually a "Local Content Plan", which includes several sub-plans (Employment Plan, Training plan, Insurance services plan, etc.), wherein the company specifies information about the impact of the company's business in the Ghanaian local economy.

The overall aim of our work is to design and implement a LegalTech application to assist the compilation and the assessment of the Local Content Plan. The present paper represents the first step of this research journey: it aims to present a first prototype of an ontology that can be used to collect and store data about companies in the upstream oil and gas industry, then used to automatically check their compliance with the L.I 2204. That is, companies are expected to use a Web interface to the Local Content Plan and determine their obligations with respect to the L.I 2204. Further data could be integrated in the ontology from other Ghanaian institutions and sources, e.g., the environmental protection agency (EPA), and double-checked against the information entered by the company. These double-checks, not implemented in our current work, are of course intended to detect (possibly unintentional) oversights and errors as well as to speed up and assist the compilation of the Local Content Plan by

[6] See https://www.gnpcghana.com/operations.html
[7] Listed at https://www.petrocom.gov.gh/laws-regulations

self-inserting the data already known. In our prototype, companies are expected to enter data about the bank supporting their financial operations, the law firm that is assisting their business, and their employees.

Some of the legal requirements that the Local Content Plan is intended to assess are the following:

(1) a. Is the EP company banking with a Ghanaian bank?
b. Is the EP company hiring the legal services of a Ghanaian law firm?
c. Is the EP company employing at least 30% of Ghanaian management staff?
d. Is the EP company employing at least 20% of Ghanaian technical core staff?
e. Etc.

We created a small ontology including some of the relevant classes and properties (TBox) from our domain. We populated the ontology with some sample individuals and relations between them (ABox). Then, we modeled some sample legal requirements, among which those in (1.a-e), as SHACL-SPARQL rules.

These rules create additional classes and properties to distinguish compliant and non-compliant individuals, and populate the classes with these individuals. By executing the rules on the (asserted) ontology, a new (inferred) ontology is obtained. The latter will therefore represent which companies comply or not with the modeled legal requirements as well as the *explanations* why they do or do not comply with these requirements.

The hypothetical LegalTech application for the Local Content Plan will query the inferred ontology via simple SPARQL queries, in order to generate a report of the compliance assessment. Note that the inferred triples are not saved and stored together with the original asserted ontology. The additional classes and properties have the sole purpose of classifying the companies as compliant or non-compliant. Once these have been identified and communicated to the LegalTech application, the inferred ontology is simply discharged.

The next two subsections illustrate part of the asserted ontology and some SHACL-SPARQL queries[8].

4.1 The (asserted) ontology

We modeled the domain of the Local Content Plan as an ontology in OWL. The ontology includes classes referring to the sets of relevant entities. Some of these classes are shown in Figure 2.

Figure 2 shows the classes of legal entities, activities, areas, and structures involved in the modeled legal requirements as well. For example, `SectorCompany` denotes the set of all companies in the upstream oil and gas industry operating in Ghana. Individuals from one class can be related to individuals from another

[8] The full ontology and list of queries is available on https://github.com/liviorobaldo/jurisin2023ca, together with Java software to execute the latter on the former thus obtaining the inferred ontology.

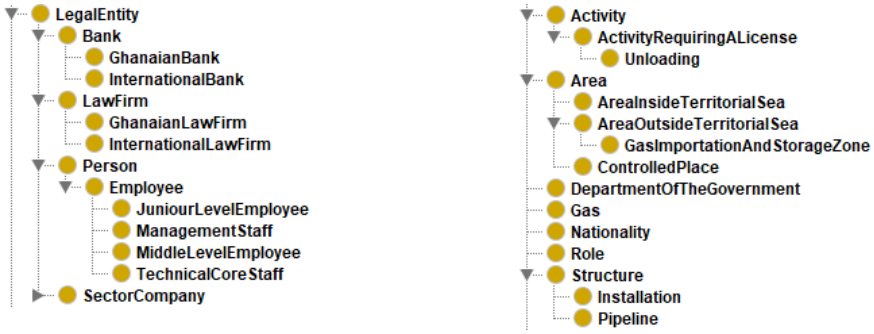

Fig. 2. Some of the classes in the ontology for the Local Content Plan

class by object properties. For instance, each individual in `Employee` is related with an individual in `Nationality` via an object property "`is`"; `Nationality` is a value partition including the individuals `ghanaian`, `italian`, `american`, etc.

On the other hand, individuals in `SectorCompany` are associated with information specifying the areas they operate, the activities they carry out in these areas, the structures used within these activities, the type of gas (`methane`, `propane`, `butane`, etc.) they work with, etc.

The ontology has been then populated with sample individuals in order to test the SHACL-SPARQL rules. Figure 3 shows some of these individuals and object properties (e.g., `employs`, `is`, `bank-with`). Two sample companies are considered: `companyc` and `companye`. The former banks with a Ghanaian bank while the latter banks with an international bank. Furthermore, `companyc` employs four technical core employees, having all Ghanaian nationality, while `companye` employs two technical core employees, having respectively Italian and American nationality.

It is then evident that `companyc` complies with legal requirements (1.a) and (1.d) while `companye` violates both of them. The SHACL-SPARQL rules described in the next section allows to infer these compliance checking results.

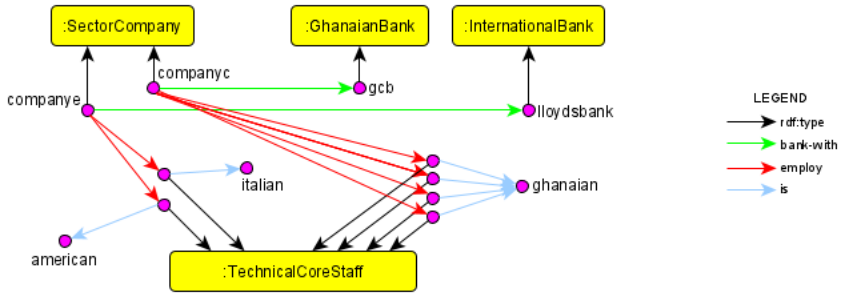

Fig. 3. Sample individuals for the case study

4.2 The SHACL/SPARQL rules

The previously referenced computational artefact includes around twenty-nine SHACL-SPARQL queries to implement some selected legal requirements from the L.I 2204. While these are all downloadable from the GitHub repository, space constraints limit showing and describing all of them in detail; we will focus only on the ones that implement (1.a) and (1.d).

The two SHACL-SPARQL rules that implement (1.a) are shown together in (2). The rules embed a SPARQL query in the form `CONSTRUCT-WHERE` within the SHACL property `sh:construct`. As specified in the `WHERE` clause, the first rule collects all companies that bank with a Ghanaian bank (`?x :bank-with ?y. ?y rdf:type :GhanaianBank.`). The `CONSTRUCT` clause:

- creates a *new* class `BLCCompliantSectorCompany` in the inferred ontology (`:BLCCompliantSectorCompany rdf:type rdfs:Class.`)
- asserts `BLCCompliantSectorCompany` as subclass of the class `SectorCompany` (`:BLCCompliantSectorCompany rdfs:subClassOf :SectorCompany.`)
- asserts the individuals `?x` that satisfy the `WHERE` clause as instances of this new class (`?x rdf:type :BLCCompliantSectorCompany.`)

The second rule in (2) is very similar to the first one. The rule collects all individuals that bank with an international bank and asserts them as instances of a newly created class `BLCNonCompliantSectorCompany`.

Thus, the two rules together distinguish individuals that comply with (1.a) from those that do not: the LegalTech application will query the inferred ontology by listing all individuals belonging to either `BLCCompliantSectorCompany` or `BLCNonCompliantSectorCompany` via simple SPARQL queries.

(2)
```
sh:rule [rdf:type sh:SPARQLRule;
    sh:prefixes[sh:declare [sh:prefix"rdf";sh:namespace"..."], ... ];
    sh:construct """
      CONSTRUCT{ :BLCCompliantSectorCompany rdf:type rdfs:Class.
            :BLCCompliantSectorCompany rdfs:subClassOf :SectorCompany.
            ?x rdf:type :BLCCompliantSectorCompany. }
      WHERE{ ?x :bank-with ?y. ?y rdf:type :GhanaianBank. }""";];

sh:rule [rdf:type sh:SPARQLRule;
    sh:prefixes[sh:declare [sh:prefix"rdf";sh:namespace"..."], ... ];
    sh:construct """
      CONSTRUCT{ :BLCNonCompliantSectorCompany rdf:type rdfs:Class.
          :BLCNonCompliantSectorCompany rdfs:subClassOf :SectorCompany.
          ?x rdf:type :BLCNonCompliantSectorCompany. }
      WHERE{ ?x :bank-with ?y. ?y rdf:type :InternationalBank. }""";]
```

Although the rules in (2) employ a different technology than [10]'s and [27]'s, the expressivity and the "modus operandi" of the three approaches is exactly the same. In other words, [10] and [27] are also designed to populate two classes such as `BLCCompliantSectorCompany` and `BLCNonCompliantSectorCompany` with all individuals that respectively comply with or not with (1.a).

By contrast, it is not possible to implement the legal requirement (1.d) via OWL classes/restrictions, as in [10], or via SHACL shapes, as in [27]. (1.d) requires to *count* both the number of Ghanaian technical core employees and the number of all technical core employees, and then to *calculate* whether the former is at least 20% of the latter. OWL "is-a" inferences and SHACL shapes are not expressive enough to query *metadata* of RDF individuals. On the contrary, SPARQL offers the desired expressivity thanks to its *aggregate functions*[9] and its *arithmetic functions*[10].

However, SPARQL alone is not enough to implement (1.d) because the two operations of counting the sets of technical core employees and calculating the proportion among these sets cannot be done via a single rule. SHACL provides the missing ingredient by allowing to *decouple* the implementation of the legal requirement into two *sequential* rules. By using SHACL-SPARQL rules as proposed here, the problem can be addressed.

The two SHACL-SPARQL rules that count the number of Ghanaian technical core employees and the total number of such employees are shown in (3).

(3)
```
sh:rule [rdf:type sh:SPARQLRule; sh:order 0;
  sh:prefixes[sh:declare [sh:prefix"rdf";sh:namespace"..."], ... ];
  sh:construct """
    CONSTRUCT {?x :gh_tec_emp ?gh_tec_emp.}
    WHERE{ SELECT ?x (count(?y) as ?gh_tec_emp)
           WHERE{ ?x rdf:type :SectorCompany.
                  ?x :employ ?y.
                  ?y rdf:type :TechnicalCoreStaff.
                  ?y :is :ghanaian.} GROUP BY ?x}"""]

sh:rule [rdf:type sh:SPARQLRule; sh:order 0;
  sh:prefixes[sh:declare [sh:prefix"rdf";sh:namespace"..."], ... ];
  sh:construct """
    CONSTRUCT {?x :tec_emp ?tec_emp.}
    WHERE{ SELECT ?x (count(?y) as ?tec_emp)
           WHERE{ ?x rdf:type :SectorCompany.
                  ?x :employ ?y.
                  ?y rdf:type :TechnicalCoreStaff.} GROUP BY ?x}"""]
```

In (3), `sh:order` is the SHACL operator to order the rules. These are executed from the lowest value of `sh:order` to the highest one. The two rules associate every sector company `?x` with, respectively, their numbers of Ghanaians technical core employees and their number of overall technical core employees via two newly created datatype properties `gh_tec_emp` and `tec_emp`.

A separate rule, shown in (4) and executed *after* the ones in (3), because its `sh:order` is equal to 1, calculates the proportion between the values of the datatype properties `gh_tec_emp` and `tec_emp`, asserted via the previous rules.

[9] https://en.wikibooks.org/wiki/SPARQL/Aggregate_functions
[10] https://en.wikibooks.org/wiki/SPARQL/Expressions_and_Functions

(4) asserts all individuals for which the proportion is lower than 20% as instances of a newly created class `Nc_Gh_Tec_Emp`.

```
(4)  sh:rule [rdf:type sh:SPARQLRule; sh:order 1;
       sh:prefixes[sh:declare [sh:prefix"rdf";sh:namespace"..."], ... ];
       sh:construct """
         CONSTRUCT{ :Nc_Gh_Tec_Emp rdf:type rdfs:Class.
                    ?x rdf:type :Nc_Gh_Tec_Emp. }
         WHERE{ ?x rdf:type :SectorCompany.
                ?x :gh_tec_emp ?gh_tec_emp.
                ?x :tec_emp ?tec_emp.
                FILTER(?gh_tec_emp<(?tec_emp*0.2)). }"""]
```

Finally, the LegalTech application can again retrieve the list of individuals belonging to the class `Nc_Gh_Tec_Emp`, i.e., the list of sector companies that do not comply with (1.d), via a simple SPARQL query.

5 Conclusions and future works

This paper contributes the means to automatise compliance checking with Semantic Web technologies. The main motivation behind researching solutions grounded on W3C standards is the hypothesis that, in the future, these standards will likely serve as the basis of symbolic explainable Artificial Intelligence, particularly for LegalTech applications.

Some recent approaches along these lines, e.g., [6] and [10], propose solutions for compliance checking based on OWL2 inferences; the main motivation behind this technological choice is to keep the framework *decidable*.

Although controlling computational complexity is of course crucial, it should be privileged over the expressivity of the inferences only when there is really no other way to make the application working in reasonable time.

This paper provided evidence that OWL2 inferences are not enough expressive for representing several compliance checks required by existing regulations, specifically those checking and aggregating *metadata* of RDF individuals.

Subsection 4.2 above exemplified this kind of checks out of a real-world legal requirement that we found in the Local Content and Local Participation Regulations for extracting oil and gas in Ghana. Companies in the oil and gas upstream industry are required to employ at least 20% of Ghanaian technical core staff. In order to represent this requirement, we had to use SHACL-SPARQL rules: SPARQL provides aggregate and arithmetic operators, while SHACL allows to build sequences or flow charts of operations by specifying priorities on the rules.

Although we have not conducted (yet) any empirical investigation about the frequency of this kind of norms in existing regulations, we believe they are rather frequent. Regulations often impose legal constraints on, for instance, *sums* of money or *minimum*/*maximal* numerical values, or they require to *count* number of days/requests/attempts/etc., etc. All these constraints require to process metadata about RDF individuals. Therefore, their implementation requires the

same expressivity offered by SHACL-SPARQL rules but not by OWL or SHACL shapes and Triple rules.

On the other hand, the computational complexity of SPARQL and SHACL does not seem to be significantly problematic (cf. [21,1]), nor, more generally, the one of compliance checkers based on sets of explicit if-then rules (cf. [30]). In fact, SHACL-SPARQL rules may be easily converted into other rule-based logical languages such as Answer Set Programming [24], for which automated reasoners with very good computational performance are available.

In future work, we will further enrich the ontology and evaluate it. Furthermore, an important direction of research is to incorporate time management in the ontology (cf. [25]), for which we plan to import existing ontologies such as OWL-Time[11] and the Time-indexed Value in Context ontology [22]. In this regard, we also intend to use a combination of SHACL and SPARQL since SPARQL vocabulary includes operators to compare dates.

On the contrary, OWL vocabulary does not include operators for comparing dates thus the formalization of time management in OWL seems to be harder, if not impossible. Therefore, we believe that our future works will further prove the main conclusion of this paper, i.e., that OWL is not expressive enough to check compliance with existing norms, many of which require to satisfy temporal deadlines or fix the maximal duration of certain permitted actions.

Indeed, the need to incorporate time management in our formalization stems from a lacuna that we found in the LI 2204: even if it is mandatory for contractors, subcontractors, licensees, and allied entities to employ a percentage of Ghanaians at certain levels of employment (management staff, technical core staff, etc.), yet a provision has not been made to define *the point in time* in which these Ghanaian employees must be employed in relation to the submission of the annual local content plan. In other words, a contractor, subcontractor, licensee, or allied entity can merely meet the LI 2204's legal requirements by employing the required percentage of Ghanaians *one day before* submitting the local content plan and by dismissing them *one day after* the submission. By playing this "trick", the legal requirement would be formally complied with, but the purposes for which LI 2204 was enacted would be obviously nullified.

The identified gap in LI 2204 could be simply solved by establishing a temporal threshold, i.e., a borderline date before which all Ghanaian employees must be employed in relation to the submission of annual local content plan. For instance, the legislation could specify that only employees hired *at least six months* prior the submission date of the local content plan can be considered.

Once the threshold has been fixed, compliance can be checked via SHACL rules that employ the SPARQL operators for comparing dates. Only if the comparison shows that a Ghanaian employee was employed before the threshold date, then the employee can count towards the required percentage.

Time management will allow us to model several further legal requirements arising from both legislation and contractual clauses. For instance, our ontology could be extended in order to associate each employee with the minimal period

[11] https://www.w3.org/TR/owl-time

(e.g., number of years) in which, by contract, s/he must be employed before s/he can be dismissed. SHACL rules will calculate the period of employment from the start and end date of the employment and compare it with the stipulated minimal employment period to check compliance with the contractual requirement.

References

1. Ahmetaj, S., David, R., Ortiz, M., Polleres, A., Shehu, B., Šimkus, M.: Reasoning about Explanations for Non-validation in SHACL. In: Proc. of 18th International Conference on Principles of Knowledge Representation and Reasoning (2021)
2. Alviano, M., Calimeri, F., Dodaro, C., Fuscà, D., Leone, N., Perri, S., Ricca, F., Veltri, P., Zangari, J.: The ASP system DLV2. In: LPNMR. Lecture Notes in Computer Science, vol. 10377, pp. 215–221. Springer (2017)
3. Athan, T., Boley, H., Governatori, G., Palmirani, M., Paschke, A., Wyner, A.: LegalRuleML: From Metamodel to Use Cases. In: Morgenstern, L., Stefaneas, P., Lévy, F., Wyner, A., Paschke, A. (eds.) Theory, Practice, and Applications of Rules on the Web. Springer Berlin Heidelberg (2013)
4. Boella, G., Caro, L.D., Humphreys, L., Robaldo, L., Rossi, P., van der Torre, L.: Eunomos, a legal document and knowledge management system for the web to provide relevant, reliable and up-to-date information on the law. Artificial Intelligence and Law **24**(3) (2016)
5. Boella, G., Caro, L.D., Rispoli, D., Robaldo, L.: A system for classifying multi-label text into eurovoc. In: Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL). ACM (2013)
6. Bonatti, P.A., Ioffredo, L., Petrova, I.M., Sauro, L., Siahaan, I.S.R.: Real-time reasoning in OWL2 for GDPR compliance. Artificial Intelligence **289** (2020)
7. Ceci, M.: Representing judicial argumentation in the semantic web. In: Casanovas, P., Pagallo, U., Palmirani, M., Sartor, G. (eds.) AI Approaches to the Complexity of Legal Systems. Lecture Notes in Computer Science, vol. 8929. Springer (2013)
8. De Vos, M., Kirrane, S., Padget, J.A., Satoh, K.: ODRL policy modelling and compliance checking. In: Fodor, P., Montali, M., Calvanese, D., Roman, D. (eds.) Rules and Reasoning - Third International Joint Conference, RuleML+RR (2019)
9. Dentler, K., Cornet, R., ten Teije, A., de Keizer, N.: Comparison of reasoners for large ontologies in the OWL 2 EL profile. Semantic Web **2**(2), 71–87 (2011)
10. Francesconi, E., Governatori, G.: Patterns for legal compliance checking in a decidable framework of linked open data. Artificial Intelligence and Law (2022)
11. Gandon, F., Governatori, G., Villata, S.: Normative requirements as linked data. In: Wyner, A.Z., Casini, G. (eds.) Legal Knowledge and Information Systems. vol. 302. IOS Press (2017)
12. Gordon, T.F.: Constructing legal arguments with rules in the legal knowledge interchange format (LKIF). In: Casanovas, P., Sartor, G., Casellas, N., Rubino, R. (eds.) Computable Models of the Law. Springer (2008)
13. Horrocks, I.: OWL: A description logic based ontology language. In: van Beek, P. (ed.) Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3709, pp. 5–8. Springer (2005). https://doi.org/10.1007/11564751_2, https://doi.org/10.1007/11564751_2
14. Humphreys, L., Boella, G., van der Torre, L., Robaldo, L., Di Caro, L., Ghanavati, S., Muthuri, R.: Populating legal ontologies using semantic role labeling. Artificial Intelligence & Law **29**(2) (2021)

15. Lahann, J., Scheid, M., Fettke, P.: Utilizing machine learning techniques to reveal VAT compliance violations in accounting data. In: IEEE 21st Conference on Business Informatics (CBI). vol. 1. IEEE (2019)
16. Leone, V., Caro, L.D., Villata, S.: Taking stock of legal ontologies: a feature-based comparative analysis. Artificial Intelligence and Law **28**(2), 207–235 (2020)
17. Nanda, R., Caro, L.D., Boella, G., Konstantinov, H., Tyankov, T., Traykov, D., Hristov, H., Costamagna, F., Humphreys, L., Robaldo, L., Romano, M.: A unifying similarity measure for automated identification of national implementations of european union directives. In: Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL). ACM (2017)
18. Palmirani, M., Martoni, M., Rossi, A., Bartolini, C., Robaldo, L.: Pronto: Privacy ontology for legal compliance. In: 18^{th} EU conference on Digital Government (2018)
19. Palmirani, M., Governatori, G.: Modelling legal knowledge for GDPR compliance checking. In: 31st Conference on Legal Knowledge and Information Systems (2018)
20. Pareti, P., Konstantinidis, G., Norman, T.J., Sensoy, M.: SHACL constraints with inference rules. In: The 18th International Semantic Web Conference (ISWC). Lecture Notes in Computer Science, vol. 11778. Springer (2019)
21. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of sparql. ACM Transactions on Database Systems **34**(3) (2009)
22. Peroni, S.: The Semantic Publishing and Referencing Ontologies. Springer International Publishing (2014)
23. Reed, W.J.: The pareto, zipf and other power laws. Economics Letters **74**(1) (2001)
24. Robaldo, L., Batsakis, S., Calegari, R., Calimeri, F., Fujita, M., Governatori, G., Morelli, M., Pacenza, F., Pisano, G., Satoh, K., Tachmazidis, I., Zangari, J.: Compliance checking on first-order knowledge with conflicting and compensatory norms - a comparison among currently available technologies. Artificial Intelligence and Law (to appear) (2023)
25. Robaldo, L., Caselli, T., Russo, I., Grella, M.: From Italian text to TimeML document via dependency parsing. In: Computational Linguistics and Intelligent Text Processing - 12th International Conference, CICLing 2011, Tokyo, Japan, 2011. pp. 177–187 (2011)
26. Robaldo, L., Pacenza, F., Zangari, J., Calegari, R., Calimeri, F., Siragusa, G.: Efficient compliance checking of rdf data. Journal of Logic and Computation (to appear) (2023)
27. Robaldo, L.: Towards compliance checking in reified I/O logic via SHACL. In: Maranhão, J., Wyner, A.Z. (eds.) Proc. of 18th International Conference for Artificial Intelligence and Law (ICAIL 2021). ACM (2021)
28. Robaldo, L., Bartolini, C., Palmirani, M., Rossi, A., Martoni, M., Lenzini, G.: Formalizing GDPR provisions in reified I/O logic: The DAPRECO knowledge base. Journal of Logic, Language, and Information **29**(4) (2020)
29. Sasu, D.: Oil and gas sector contribution to GDP in Ghana 2014-2024. In: Statista, available at https://www.statista.com/statistics/1235708/gdp-of-the-oil-and-gas-industry-in-ghana (2021)
30. Sun, X., Robaldo, L.: On the complexity of input/output logic. The Journal of Applied Logic **25**, 69–88 (2017)
31. Zhang, R., El-Gohary, N.: A machine learning approach for compliance checking - specific semantic role labeling of building code sentences. In: Proc. of the 35th International Conference of IT in Design, Construction, and Management (2018)